My problem # 1: bridge between logic and neural

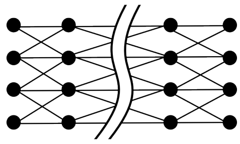
甄景贤 (King-Yin Yan)

General.Intelligence@Gmail.com

1 神经网络的结构

一个 神经网络 F 基本上是：

$$\boldsymbol{y} =$$



$$(\boldsymbol{x})$$

$$F(\boldsymbol{x}) = \sigma(W_1 \sigma(W_2 ... \sigma(W_L\ \boldsymbol{x}))) \tag{1}$$

为简单起见，输入 $\boldsymbol{x}$ 和输出 $\boldsymbol{y}$ 都是 dim-n vector。
L 是层数，W_ℓ 是每层的**权重**矩阵 $(= n \times n$ matrix)。
σ 是 sigmoid function （其作用是赋予非线性）：

$$\sigma(\xi) = \frac{1}{1 + e^{-\xi}} \tag{2}$$

applied **component-wise**。必要时 σ 可以换成 $\mathbb{C}$ 上的 polynomials。

可以证明，如果层数和 neuron 个数足够多，这些 F functions 的 family $\mathcal{F}$ 在某个 function space 上是 dense 的。

Machine learning 的目的是在这 function space 上「计分」(ie, distribute scores over functions F).

可以将 $\mathcal{F}$ 看成是由 σ 和 W's generate 出来的 Banach algebra （乘法是 composition）。但我不懂计算 $\mathcal{F}$ 的 basis（也不知计出来有没有用）。

2 逻辑结构

重点是想在 $\mathcal{F}$ 的结构上「加入」逻辑 rules 的结构。逻辑是指由一些命题推导出另一些命题：

$$A, B, C, ... \vdash D \tag{3}$$

这里的 $\vdash$ 的作用 corresponds to F。

逻辑结构有两部分：

- **Matching** structure: the state variable $\boldsymbol{x}$ is the Cartesian product of a number of smaller states $\boldsymbol{x}_i$. "Matching" means to check if $\boldsymbol{x}_i$ resides in certain polytope regions:

$$\begin{aligned} \text{if } A\boldsymbol{x}_i \geq \boldsymbol{b} \ \text{ then } \ & \mathbf{1} \\ \text{else } \ & \mathbf{0} \end{aligned} \tag{4}$$

  这个 matching 的结果要 multiply with 下面的 linkage function。

- **Linkage** structure: Consider this logic formula and notice the linkages between variables on the left and right sides:

$$\text{grandfather}(X,Z) \longleftarrow \text{father}(X,Y) \wedge \text{father}(Y,Z) \tag{5}$$

  In the state space $\mathbb{X} \ni \boldsymbol{x}$, linkage means that F projects the i-th coordinate of $\boldsymbol{x}$ directly to the j-th coordinate of $\boldsymbol{y}$. In other words,

$$f : (x_1, ...x_i, ...x_n) \mapsto (y_1, ..., (y_j = x_i), ...y_n) \tag{6}$$

  I use the notation $F_{(i,j)}$ to denote that F contains the linkage from coordinate i to coordinate j.

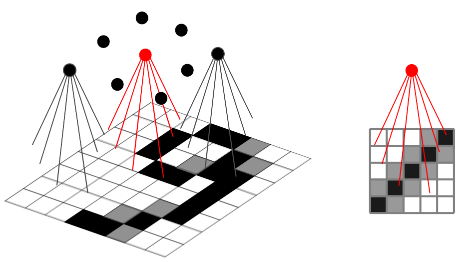
我的问题是：怎样将以上的两种 logic structure 「添加」到神经网络的 F 里。

神经网络的 F's 已经在 function space 是 dense 的。但全体的有 linkage 的函数不是 dense 的。但它是不是一个 simply connected subset？

注意：如果 F 有 linkage，那就变成一些 equational constraints over W. 但我不懂怎样将神经网络的 F 表示成有 linkage 的。

现在思考一下，神经网络怎样识别模式，或许会有帮助....

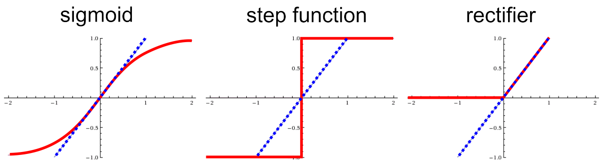
考虑最简单的情况，例如提取数字 "9" 的特徵的一层网络。这层网络可以有很多神经元（左图），每个神经元局部地覆盖输入层，即所谓视觉神经元的 local receptive field（右图）。



(7)

假设红色的神经元专门负责辨识「对角线」这一特徵。它的方程式是 $\boldsymbol{y} = \sigma(W\boldsymbol{x})$。矩阵 W 的作用是 affine「旋转」特徵空间，令我们想要的特徵指向某一方向。然后再用 σ 「**挤压**」想要的特徵和不想要的特徵。Sigmoid 之后的输出，代表某类特徵的**存在与否**，即 $\{0, 1\}$。这是一种资讯的**压缩**。

讲一点 chaos theory：$^{-1}$ 的作用是「扯」(stretch)，将本来邻近的两点的距离非线性地拉远。看看以下各种常见的激活函数，它们全都是相对於 identity $y = x$ 的非线性 deformation：



(8)

这和 Steven Smale 提出的「马蹄」[?] 非常类似，它是制造混沌的处方之一。Smale 马蹄的另一个变种叫做 baker map，其作用类似於「搓面粉」。换句话说，「拉扯」然后放回原空间，如此不断重复，就会产生混沌 [?] [?]。（神经网络的时间逆向就是 $^{-1}$，所以时间向前也是混沌。）

这里有一点重大意义：$\boldsymbol{F}^{-1}$ 有混沌的典型特徵：它「对初始状态的微少变化非常敏感」。换句话说 $\boldsymbol{F}$ 的逆是 unpredictable 的，简言之，神经网络 $\boldsymbol{F}$ 是**不可逆**的压缩过程。

最近一个有趣的例子是 DeepDream [?]，它用神经网络的 $\boldsymbol{F}^{-1}$ 产生有迷幻感觉的 pre-image，证实了 $\boldsymbol{F}^{-1}$ 可以和原本的 image 差距很大：

(9)

总括以上，可以归结出一些原则：

- 每个神经元的输出代表某个 feature 的存在与否
- 更高层的神经元代表下层 features 之间的**关系**

(10)

这些原则暂时未能严格地表述和证明，或者可以叫它做 **神经原理 (Neural Postulate)**。

凭这个思路推广，可以推测这样的 correspondence：

$$
\begin{array}{rccl}
 & \mathbb{M} & \simeq & \mathbb{X} \\
\text{逻辑物体} & & \Leftrightarrow & \text{neuron} \\
\text{逻辑关系} & & \Leftrightarrow & \text{relation between higher and lower neurons}
\end{array}
\tag{11}
$$

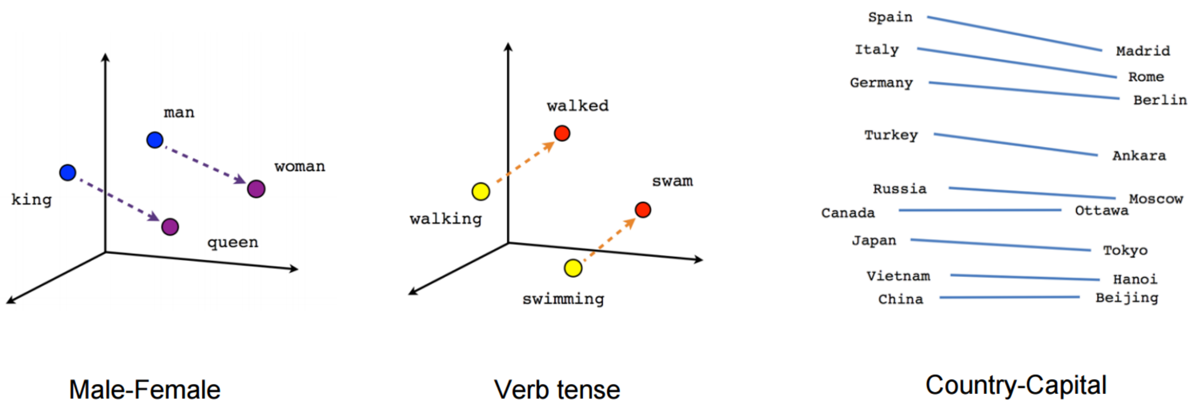
3 逻辑的结构

一个**逻辑系统**可以这样定义：

- 一些 constant symbols，predicate symbols，和 function symbols
- 由上述的原子建立 **命题** (propositions)
- 命题之间可以有连接词：$\neg, \wedge, \vee$ 等
- 建立 **逻辑后果** (consequence) 关系：$\Gamma \vdash \Delta$

3.1 Semantic distance 与 Compositionality

我们的目的是想将整套逻辑 AI 的器材搬到 连续 的微分流形上去实现。这个做法，是受了 Google 的 Word2Vec [?] 算法的启发，它可以将自然语言的词语 embed 到度量空间中，而且词语之间的距离是 semantic distance （**语义学上的距离**）：



(12)

当然，word2vec 一经发表之后，很多人开始构思怎样把「句子」也 embed 到度量空间上。第一个想法当然是用 **tensor product**，例如 “he loves her” 这句子就变成了 he $\otimes$ loves $\otimes$ her。

但这个做法很有问题；在语言学 / 逻辑学里有一个重要概念叫 **compositionality**，它说：

$$\underline{\text{合成物的语义 (semantics) 由其所构成的原子生成，而不依赖於其他资料}} \tag{13}$$

例如「我爱你」由「我、爱、你」三个原子组成。如果原子用向量表示，那么「我爱你」和「没有你我不能生存」这两句意义相近，但形式上 (syntax) 却很不同，导致这两组原子的 tensor products 可能相距颇远。

用 tensor product 的做法，得出的空间结构是 syntactic 而不是 semantic 的，但神经网络学习的重点是 泛化 (generalization)，它基於「邻近的点的意义相近」才能成功，所以一定要 semantic distance。

3.2 「二分法」逻辑

上节看到 tensor product 的做法不行，它用原子 “algebraically” 组合成句子，原子之间的距离可以是语义学相近的，但「乘出来」的句子未必有 semantic distance。我们想要的是相反的特性：句子之间要有 semantic distance，但这样似乎不能将句子分拆成原子，尤其是日常语言中的词语 (words)。

暂时只考虑一个 thought （= 命题）如何表示。

假设思维空间的 metric 是语义的（语义相近则点的距离相近）。根据「神经原理」(10)，每个 thought 必然是由一组 features 表述，例如 $\boldsymbol{x} = (x_1, x_2, ..., x_n)$，这里每个 x_i 就好比一个概念原子。但问题是我们不想 thought 用原子「乘出来」，怎办？

记得逻辑学历史中 George Boole 曾经提出过一种逻辑 [1]，例如 x 代表「是人的」，y 代表「会死的」，那么「所有人都会死」就是：

$$x\,(1-y)=0 \tag{14}$$

这里 x,y 是 “classes”，乘法是集合的 intersection，即 $\cap$。这种乘法和上面描述的句子 / 词语乘法不同；首先，这乘法是 commutative 的，$x\cap y=y\cap x$。如果将一个 thought 表示为：

$$\boldsymbol{x}=x_1,x_2,...,x_n=x_1\cap x_2...\cap x_n \tag{15}$$

则这些 x_i 可以看成是将所有思维的空间用「二分法」切割，每个 x_i 是一个二分切割。

两个 thoughts 之间的距离可以用 Hamming distance 量度，它是一个 semantic distance，尤其是当 n 比较大时，可能是一个效果不错的 metric。

总括来说：「二分法」逻辑用一组 binary features 去定义一个 thought，一条思维是 n-维 hypercube 上的一个顶点。

3.3 逻辑中的 linkage 现象

在经典逻辑中有一个”linkage” 的现象，例如以下这个式子：

$$\text{grandfather}(X,Z)\longleftarrow\text{father}(X,Y)\wedge\text{father}(Y,Z) \tag{16}$$

那意思是说，无论右边的变量（例如 X）怎样改变，左边的 X 必须代入相同的值。这是 代入 (substitution) 的本质。

简单来说，需要用一些 functions 描述这些逻辑式子。当然，可以用完全无限制 (free) 的函数，但我不想丧失结构（因为这些函数的整个家族都有此 linkage 结构，每次重复学习同一种结构是浪费时间的）。

虽然 (16) 是 first-order predicate logic，我们可以用类似的方法泡制「二分法」逻辑中的 linkage，其功效或许差不多吧？

换句话说，命题就是 thought，一个 conditional formula 是一个函数，它将几个（前提）命题映射到一个（结论）命题。在这函数里面有 linkage 结构，这 linkage 结构是 positional 的，亦即是说，一个 linkage 是由 source 空间的第 p 命题的第 q 个分量到 target 空间的第 r 个分量的 projection function π。

一个没有 linkage 的函数 f 可以「投影」到某个 projection function 变成有 linkage：

$$f_r\mapsto\pi_{p,q} \tag{17}$$

但这只是一条 rule，rules 构成 KB，但似乎 $\boldsymbol{F}$ 应该包含整个 KB。

[1] 这和后世为他命名的 Boolean logic 不同。

3.4 用 relation algebra 如何？

以前曾经对 relation algebra [?] [?] 有些憧憬，因为它比较接近人类自然语言，但其实 relation algebra (RA) 和 first-order logic (FOL) 基本上是等效的，在 FOL 里面有 linkage 的复杂性，但在 RA 里面这个复杂性其实也没有消失。可以说「复杂度是守恒的」。

举例来说，在 (16) 中表达「爸爸的爸爸是爷爷」，可以用 RA 更简单地表达：

$$\text{father} \circ \text{father} = \text{grandfather} \tag{18}$$

实际的推导是这样的：

$$\text{John father Pete} \tag{19}$$
$$\text{Pete father Paul} \tag{20}$$
$$\text{John father} \circ \text{father Paul} \tag{21}$$

这时要**代入**上面的等式才能得出结论

$$\text{John grandfather Paul} \tag{22}$$

所以 linkage 的复杂性变成了代数 formula 长度的复杂性。

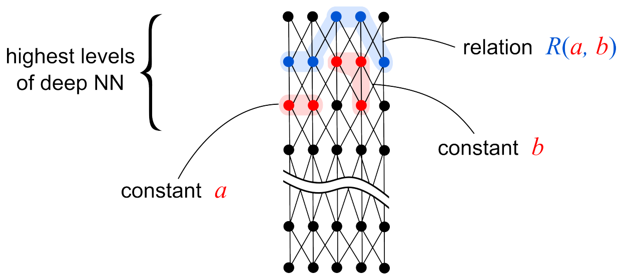
3.5 命题的集合

神经的状态空间由一些 **thoughts**（思维）组成，例如「我正在上课 $\wedge$ 邻座的女孩很漂亮」，这两个 thoughts 是独立的。也可以有另一个状态：「我正在搭地铁 $\wedge$ 邻座的女孩很漂亮」。Thoughts 独立的好处是表述的 economy。思维状态 $\boldsymbol{x}$ 就是 M 个 thoughts 的集合，M 是 working memory 的大小。认知科学里有个说法：the size of human working memory is approximately 7 ± 2 items.

思维空间是 M 个 hypercube 的卡积。

4 结论

但要注意的是这对应未必是一对一的，可能是一个 constant 对应几个 neurons 的（线性？）**组合**。具体情况可能像以下的示意图（实际上每层神经网络可能有很多神经元）：



(23)

$R(a,b)$ 可以在 a,b 的 common parents 中寻找（例如那些蓝色神经元，$R(a,b)$ 的值 = 蓝色神经元的某个线性组合）。验证的方法是：当 a 和 b 的信号都是「有」时，$R(a,b)$ 的值也应该是 true。

这篇论文并不太成功，因为跳到 (11) 和 (23) 的结论没有严谨的根据，只是直观上觉得有可能。理论上来说，既然知道了 $\mathbb{M}$ 那边是怎样生成的、$\mathbb{X}$ 那边是怎样生成的，则要在两边建立「高速公路」应该是可行的。实际上，似乎只要建立一个深度网络就可以，因为神经网路是 universal function approximator，根本不用考虑 $\mathbb{M}$ 和 $\mathbb{X}$ 这两个结构之间的关系。

进一步的研究，希望数学专业的人能帮助一下：

1. 在逻辑那边，可不可以转换成 algebraic geometry 的结构？即是说：逻辑式子 $\simeq$ 代数方程。这种代数逻辑的做法，我暂时只知道有 [?]，是很偏门的研究。
2. 能不能根据 $\mathbb{M}$ 和 $\mathbb{X}$ 的结构，找出它们之间的桥的最简单形式？可以用数学归纳法，逐步考虑 $\mathbb{M}$ 和 $\mathbb{X}$ 生成的方式，或许有帮助？

应用：对於用深度学习做 natural language understanding 的人，这理论或许会很有用。

5 Prior art

- Bader, Hitzler, Hölldobler and Witzel 在 2007 年提出了一个 neural-symbolic integration 的做法 [?]。他们首先由 logic theory 生成抽象的 Herbrand model[2]，再将 Herbrand model 映射到某个 fractal 空间，然后直接用神经网络学习那 fractal 空间。虽然用了 model theory，但他们没有利用到本文所说的 $\mathbb{M}$ 和 $\mathbb{X}$ 之间的关系。
- Khrennikov 在 1997 年开始的多篇论文中提出了用 p-adic 代数来模拟思维空间 $\mathbb{X}$ 的结构，详见 [?] 一书。一个 p-adic 数可以看成是一个 p 进制的小数，p 是任何质数。
- 经典逻辑是二元逻辑，近代已经有无数将它扩充到 fuzzy 或 probablistic 的尝试（作者也提出过 [?]），但仍未有统一的理论。与此不同的另一个方向，如果将点看成是 first-order objects，谓词是点空间上的函数，直接得到 metric structures 上的连续逻辑 (continuous first-order logic) [?]，这可以看成是一种 $\mathbb{M}$ 的结构。
- 模型论中有（超滤子）ultra-filter 和 ultra-product 这些建构，它们起源於泛函分析，最近有很多横跨模型论和 Banach 空间的新研究 [?]。简单地说 ultra-product 用来将一些 models 构造出新的乘积 models。但我粗略地看过一下之后发现 ultra-product 通常涉及无穷集合，而且是很大的物体，在计算机上应用似乎不太实际。

[2] Herbrand model 是邏輯 AI 中常用的概念，大意是用邏輯語言 $\mathcal{L}$ 生成「所有可以代入的東西」(instantiating whatever that can be instantiated)，由此產生的不含變量的句子 (sentence) 的集合。換句話說，Herbrand model 的特點是它只靠 $\mathcal{L}$ 自身產生它的模型，而不依賴任何外在結構。每个邏輯 theory 都必然至少有一个 Herbrand model。